Amendments to the Specification:

Please replace paragraph [0039] with the following amended paragraph:

[0039] Additional aspects and advantages ~~of this invention~~ will be apparent from the following detailed description ~~of preferred embodiments thereof~~, which proceeds with reference to the accompanying drawings.

Please replace paragraph [0110] with the following amended paragraph:

[0110] FIG. 2 is component 100 further including a first coordination interface 200, a second coordination interface 202, and a third coordination interface ~~204~~ 206. Coordination-centric design's components 100 provide the code-sharing capability of object-oriented inheritance through copying. Another aspect of object-oriented inheritance is polymorphism through shared interfaces. In object-oriented languages, an object's interface is defined by its methods. Although coordination-centric design's actions 104 are similar to methods in object-oriented languages, they do not define the interface for component 100. Components interact through explicit and separate coordination interfaces, in this figure coordination interfaces 200, 202, and ~~204~~ 206. The shape of coordination interfaces 200, 202, and ~~204~~ 206 determines the ways in which component 100 may be connected within a software system. The way coordination interfaces 200, 202, and ~~204~~ 206 are connected to modes 102 and actions 104 within component 100 determines how the behavior of component 100 can be managed within a system. Systemwide behavior is managed through coordinators (see FIG. 4B and subsequent).

Please replace paragraph [0118] with the following amended paragraph:

[0118] The coordination-centric design methodology provides an encapsulating formalism for coordination. Components such as component 100

interact using coordination interfaces, such as first, second, and third coordination interfaces 200, 202, and ~~204~~ 206, respectively. Coordination interfaces preserve component modularity while exposing any parts of a component that participate in coordination. This technique of connecting components provides polymorphism in a similar fashion to subtyping in object-oriented languages.

Please replace paragraph [0283] with the following amended paragraph:

[0283] FIG. 11 shows a combined coordinator 1100 with both preemption and round-robin coordination for controlling access to a resource, as discussed above. With reference to FIG. 11, components 1102, 1104, 1106, 1108, and 1110 primarily use round-robin coordination, and each includes a component coordination interface 1112, which has a component arbitrated control port 1114, a component input message port 1118, and a component output message port 1116. However, when a preemptor component 1120 needs the resource, preemptor component 1120 is allowed to grab the resource immediately. Preemptor component 1120 has a preemptor component coordination interface 1122. Preemptor component coordination interface 1122 has a preemptor arbitrated state port 1124, a preemptor output message port 1126, and a preemptor input message port 1128.